Grundläggande Programmering

Föreläsning 7 - Objektorienterad programmering

Idag

- Syntax
- Motivation
- Historien bakom objektorientering
- Klasser
- Objekt
- Attribut
- Specifika Metoder för CRUD: __init__, __repr__, __str__
- Metoder generellt
- Ryggsäck
- Bank
- Fyrverkerier!

Motivation för objektorienterad programmering

- Records: Istället för enskilda variabler, ha ett record för värden som hör ihop: exempel: Användarnamn, lösenord, hemkatalog
- Är ett enskilt värde eller en kombination av värden giltiga eller rimliga? Sätt upp en klass och se till att skapandet och uppdaterandet av data håller sig rimligt och giltigt!
- Minskad koduppprepning: Flera klasser kommer att likna varandra, låt dessa ligga i en eller flera arvshierarkier. I Grundläggande programmering ingår arvet från Object och arvet från Frame i TKInter.

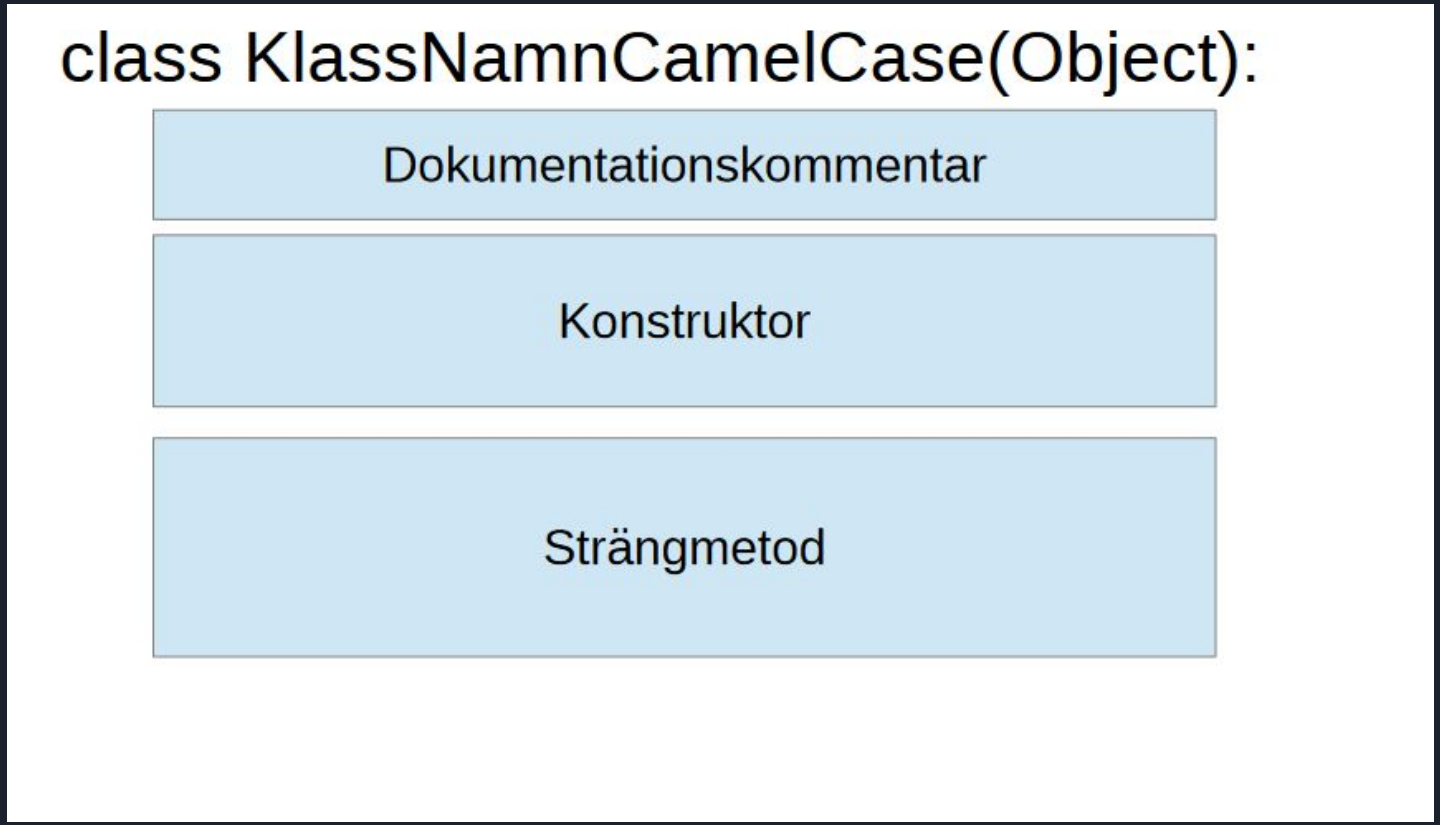
Ytterligare motivation

- Populärt: C++, C#, Java, Javascript, Typescript, Python, PHP (men inte C eller CSS) stödjer OOP.
- Definiera egna datatyper
- Dela upp ett program/problem/data i ansvarsdelar

Alternativ till OOP

- Enklare: Record syntax, struct
- Komponentbaserad programmering
- Avancerat: Entity Component System

Syntaxen för att skapa en klass



Syntaxen för en metod

```
def metod_namn(self, parameter_lista…):
```

Dokumentationskommentar

Metoder påminner om funktioner, men de har tillgång till objektets inre delar.
De känns snabbt igen på att första parametern…
...är self

Historien bakom objektorienterad programmering (OOP)

På 40- och 50-talet fanns inga objektorienterade programmeringsspråk! De rådande 4 paradigmerna i programmeringens barndom var det imperativa, det strukturerade, det procedurella och det funktionella.

På slutet av 50-talet så började arbetet med att skapa Cobol. Cobol hade från början inte objektorientering, men det hade records, som var föregångaren till klasser och objekt.

På 60-talet kom Simula som var det första objektorienterade språket. I Simula fanns arv, inkapsling, polymorfism och abstraktion, men de tre senare koncepten kallades inte för detta på 60-talet.

På 70-talet kom språket Small Talk som vidareutvecklade och populariserade konceptet.

Historien bakom dictionaries

Fanns inte i programmeringens begynnelse!

På 1960-talet fick Lisp associativa arrayer

Dessa vidareutvecklades till hashtabell, map och dictionary (olika namn för samma sak, men map är också en funktion så Pythons val att kalla datastrukturen för dictionary är pedagogiskt guld!)

På 70-talet raffinerades hashtabellerna ytterligare.

Från 80-talet och framåt ingår hashtabeller i alla grundkurser i programmering.

I fortsättningskurserna kommer ni att utvärdera olika hashfunktioner och olika kollisionshanteringssystem, men i grundkursen räcker det med att använda dem.

C och C++

Bjarne Stroustrup som arbetade på Bell Labs började utveckla C med klasser 1979.

1980 kom den första implementationen, skriven av Stroustrup själv.

1983 bytte språket namn till C++

1985 Stoustrup släppte boken The C++ Programming Language

1991 En internationell standardkommitté formades för C++.

1998 Första ISO-standarden för C++ publiceras.

2003, 2011, 2014, 2017, 2020 kom nya C++-standarder.

2023 i december publiceras nästa standard med bland annat korutiner.

C++ har lagt till multipelt arv till OOP, vilket också stöds av Python. Andra språk som Java och C# har valt att inte ta efter denna språkegenskap.

Eiffel - Design by contract

Eiffel introducerade konceptet med invarianter. Koden ska hålla enskilda datapunkter och kombinationer av datapunkter inom vissa regler och specifikationer.

Lite teori bakom objektorientering

Mängder

Abstraktioner

Arv och arvshierarkier

Minska koduppprepning

Göra data i minnet begripligt (men först måste man förstå objektorientering)

CRUD (Create, Read, Update, Delete)